A Real Time Automated Microclimate Ecosystem

Student Author: Kayce Conville
Faculty Advisor: Dr. William Slaton
University of Central Arkansas Department of Physics & Astronomy

ABSTRACT

The goal of the project is to create an automated microclimate which recreates any given ecosystem in real time. The ecosystem is controlled with a Rasberry Pi and the program is written in Python. The user is able to give the program a given weather station ID associated with Weather Underground, which the program then pulls the HTML code from the website providing the program with the real time weather information for that particular station. The program than processes the data and pulls the information wanted, like current temperature, solar radiation, hourly precipitation, and daily precipitation. The focus of the project as of now is mimicking solar radiation. The environment will be equipped with a semiconductor photodiode sensor which will give off a voltage proportional to the amount of energy given off by the light source. To create the scale of voltage versus light, data is collected with the photodiode sensor in the vicinity of a weather station which reports solar radiation. The data from the weather station and the data from the sensor can then be analyzed to create this scale. With this information, the program will then be able to take in the live data from the weather station and adjust a light source to give off the correct amount of energy, matching the real time conditions. By creating a real time automated microclimate ecosystem, it can allow the user to recreate an environment and observe the subject that is being studied in its natural habitat from anywhere in the world.

Components

The following software and components are used to collect data.

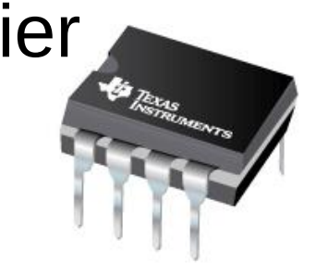
Hardware:

RasberryPi 3

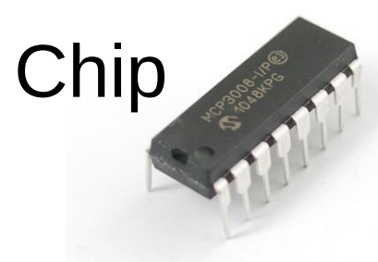
VTB Photodiode

CdS Photoresistor

LMC6942 – Operational Amplifier



MCP3008 – ADC Chip



LM385 – Voltage Reference Diode

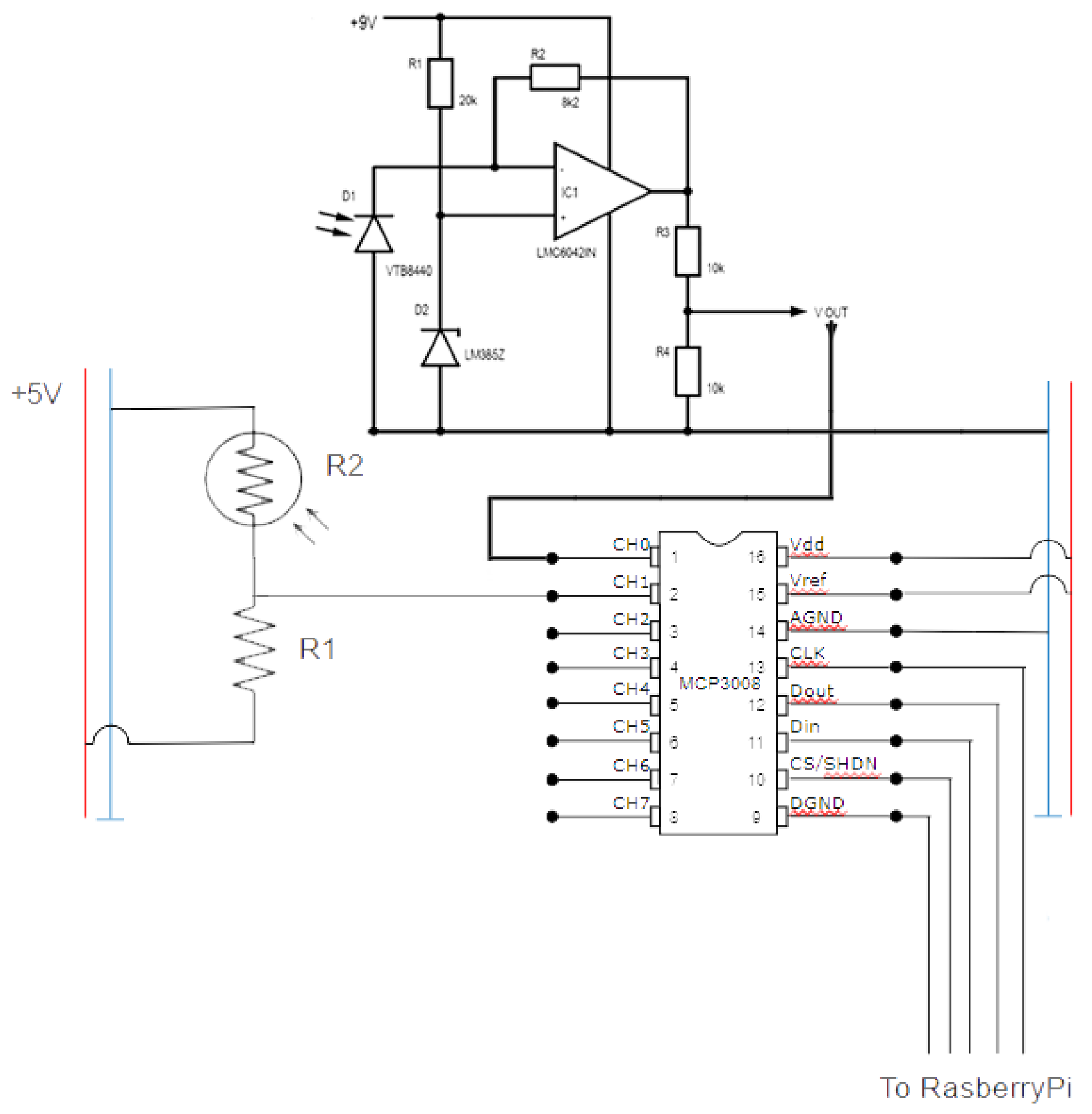
Software:

Python

Sensor Circuit

The light intensity is measured with a photoresistor within a voltage divider circuit and a photodiode. As the light intensity increases, the voltage across the photoresistor decreases and the voltage across the photodiode increases. Figure 1 depicts the circuit that is used to measure light intensity from both sensors.

Figure 1



The python code function, readADC_channel_0 in Figure 4 gives the Rasberry Pi the ability to convert the analog input to a digital input. The MCP3008 chip is a 10-bit chip (2^{10}). This gives a level of 1024 integer values. The function calc_volts_0 converts the integers given by the ADC chip into a value of voltage input.

The code is executed within the same time interval the live data is collected. Due to the fact that the photodiode reduces its resistance as the light intensity increases, the collected data is inversely proportional to the voltage data from the photodiode.

Figure 4

```
def readADC_channel_0():

    d0 = ''

    GPIO.output(CS, False)

    GPIO.output(DIN, True)

    GPIO.output(CLK, False)
    GPIO.output(CLK, True)
    GPIO.output(CLK, False)


    din_control = '1000'
    for n in din_control:
        if n == '1':
            GPIO.output(DIN, True)

        else:
            GPIO.output(DIN, False)

        GPIO.output(CLK, False)
        GPIO.output(CLK, True)
        GPIO.output(CLK, False)

    GPIO.output(CLK, False)
    GPIO.output(CLK, True)
    GPIO.output(CLK, False)

    for n in range(0,10):
        GPIO.output(CLK, False)
        GPIO.output(CLK, True)
        GPIO.output(CLK, False)

        DOUT_state = GPIO.input(DOUT)
        if DOUT_state == True:
            d0 = d0 + '1'

        else:
            d0 = d0 + '0'

    GPIO.output(CS, True)
    GPIO.output(DIN, False)

    return d0



def calc_volts_0(d0):

    d0_int = int(d0,2)
    volts_0 = 5.0*d0_int / 1023
    volts_0 = round(volts_0, 6)

    return volts_0
```

Future Improvement

The project at hand allows for many improvements. The ultimate goal is to create an automated microclimate ecosystem. Weather Underground provides many weather variables from locations all across the globe. By utilizing the data that is available it is possible to mimic all variables. On the top of the list of improvements is the addition of temperature sensors and a heat lamp to be able to mimic the actual temperature changes in real time.

ACKNOWLEDGEMENTS

- Weather Underground – weatherunderground.com
- University of Central Arkansas Department of Physics and Astronomy
- K. A. Nicoll, R.G. Harrison. Balloon-borne disposable radiometer for cloud detection. Department of Meteorology, University of Reading. 21 February 2012

Real Time Data Collection

Weather Underground is used to collect real time data for the selected area. The data collected was from a weather station located in Conway Arkansas that reports solar radiation (W/m^2).

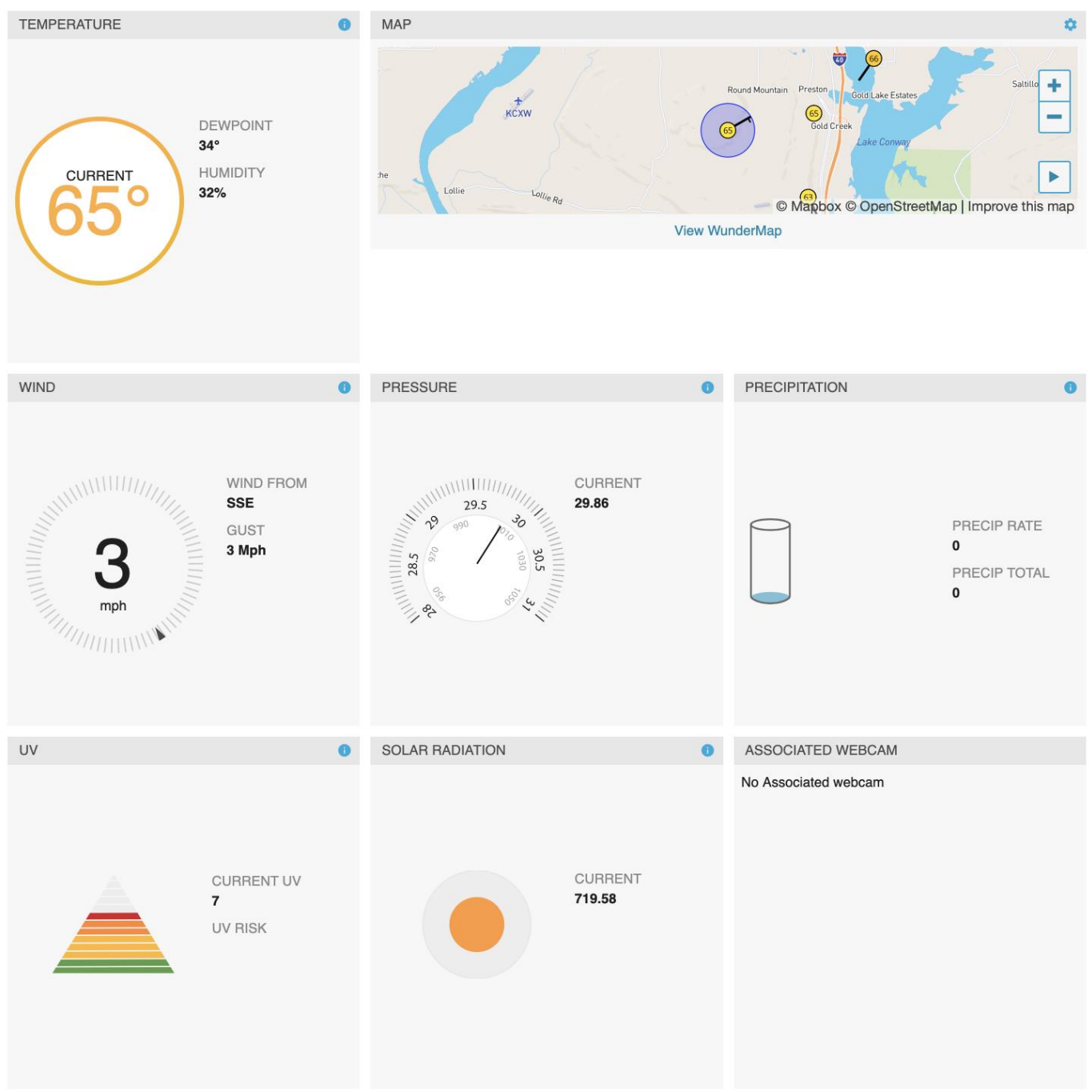
The user is able to input a specific weather station into the code, in this case it is the local station KARCONWA57. The program is then able to pull the html code associated with the webpage and process the information to pull out the selected data for a given time interval. In this case solar radiation data is being collected but as you can see in Figure 2 that there are many other variables that can be utilized.

The code in Figure 3 below depicts how BeautifulSoup, a web scraping package, is used to collect the data.

Figure 3

```
    #opens webpage
url1 = urllib.request.urlopen(webpage1)

content1 = url1.read()

    #gets text from page
soup1 = BeautifulSoup(content1, "html.parser")

title1 = soup1.get_text()

    #writes text from page into a .txt file
file = open("Weather_Forecast_weather_underground"+str(check)+".txt",'wb')
file.write(unicodedata.normalize('NFKD', title1).encode('ascii','ignore'))
file.close()

    #searches the file for the solar radiation data
searchfile = open("Weather_Forecast_weather_underground"+str(check)+".txt", "r")
x = 0
for line in searchfile:
    if x < 1:
        if "solarradiation" in line:
            solarradiation_raw = line
            a = solarradiation_raw.find(':')
            b = solarradiation_raw.find(',')
            solarradiation = line [a+2:b]
            solarradiation_data.append(solarradiation)
            x +=1

searchfile.close
```

Figure 2



TEMPERATURE

CURRENT 65°

DEWPOINT 34°

HUMIDITY 32%

MAP

© Mapbox © OpenStreetMap | Improve this map

View WunderMap

WIND

3 mph

WIND FROM SSE

GUST 3 Mph

PRESSURE

CURRENT 29.86

PRECIPITATION

PRECIP RATE 0

PRECIP TOTAL 0

UV

CURRENT UV 7

UV RISK

SOLAR RADIATION

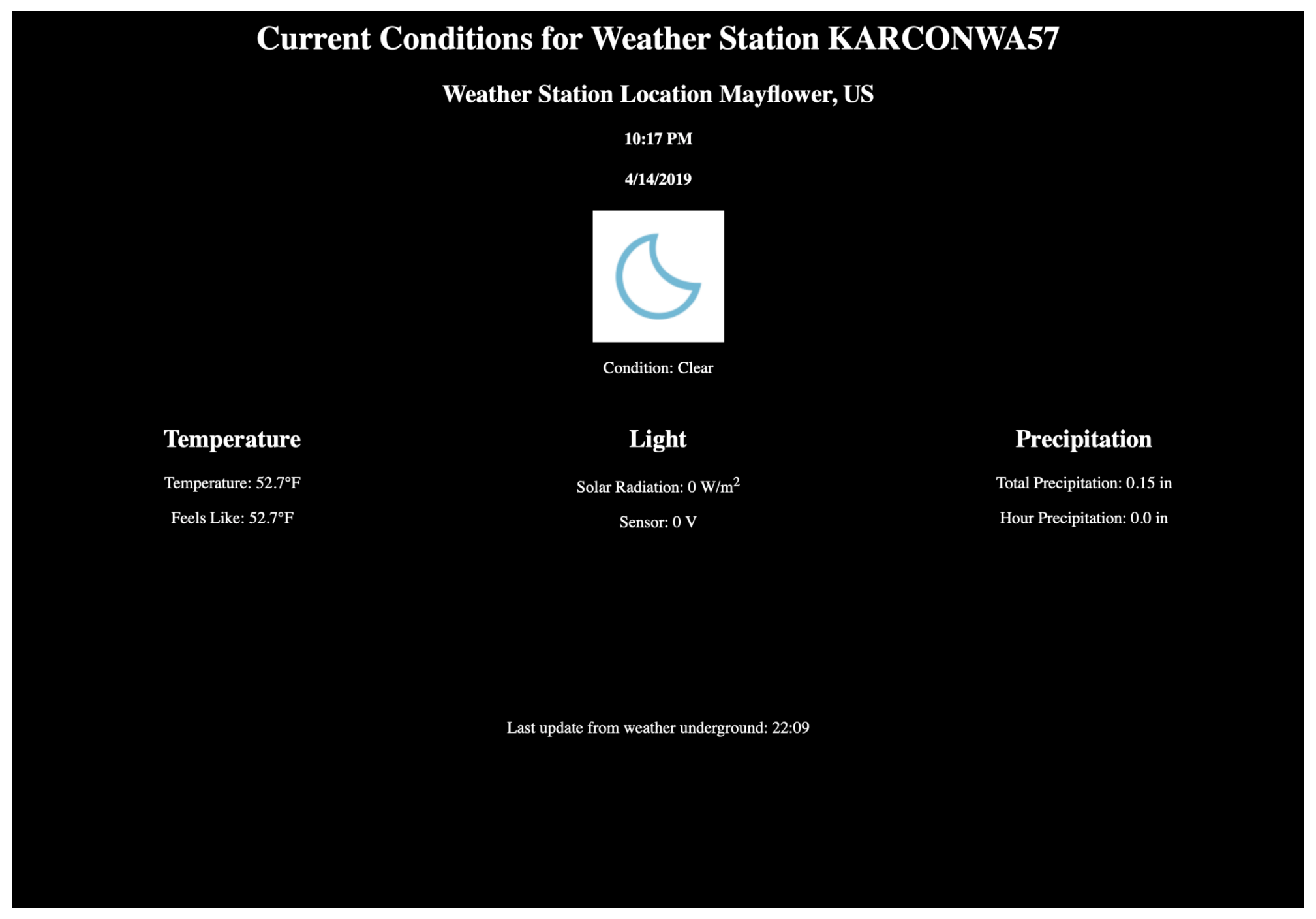
CURRENT 719.58

ASSOCIATED WEBCAM

No Associated webcam

The collected data from both weather underground and the photodiode sensor are then transferred into a graphical user interface which updates every time the information is collected as seen in figure 7.

Figure 7



Current Conditions for Weather Station KARCONWA57

Weather Station Location Mayflower, US

10:17 PM

4/14/2019

Condition: Clear

Temperature

Temperature: 52.7°F

Feels Like: 52.7°F

Light

Solar Radiation: 0 W/m^2

Sensor: 0 V

Precipitation

Total Precipitation: 0.15 in

Hour Precipitation: 0.0 in

Last update from weather underground: 22:09

Collected Data

The data below in Figure 5 shows how the photodiode voltage responds to light. There is a clear correlation between the photodiodes response and the measured solar radiation. Figure 6 shows a photoresistors response to light for the same day and time intervals as figure 5. Since the photoresistor is within a voltage divider circuit the response is inversely related to the measured solar radiation. It is also visible in figure 6, that the photoresistor does not have the same sensitivity to light as the photodiode.

Figure 5

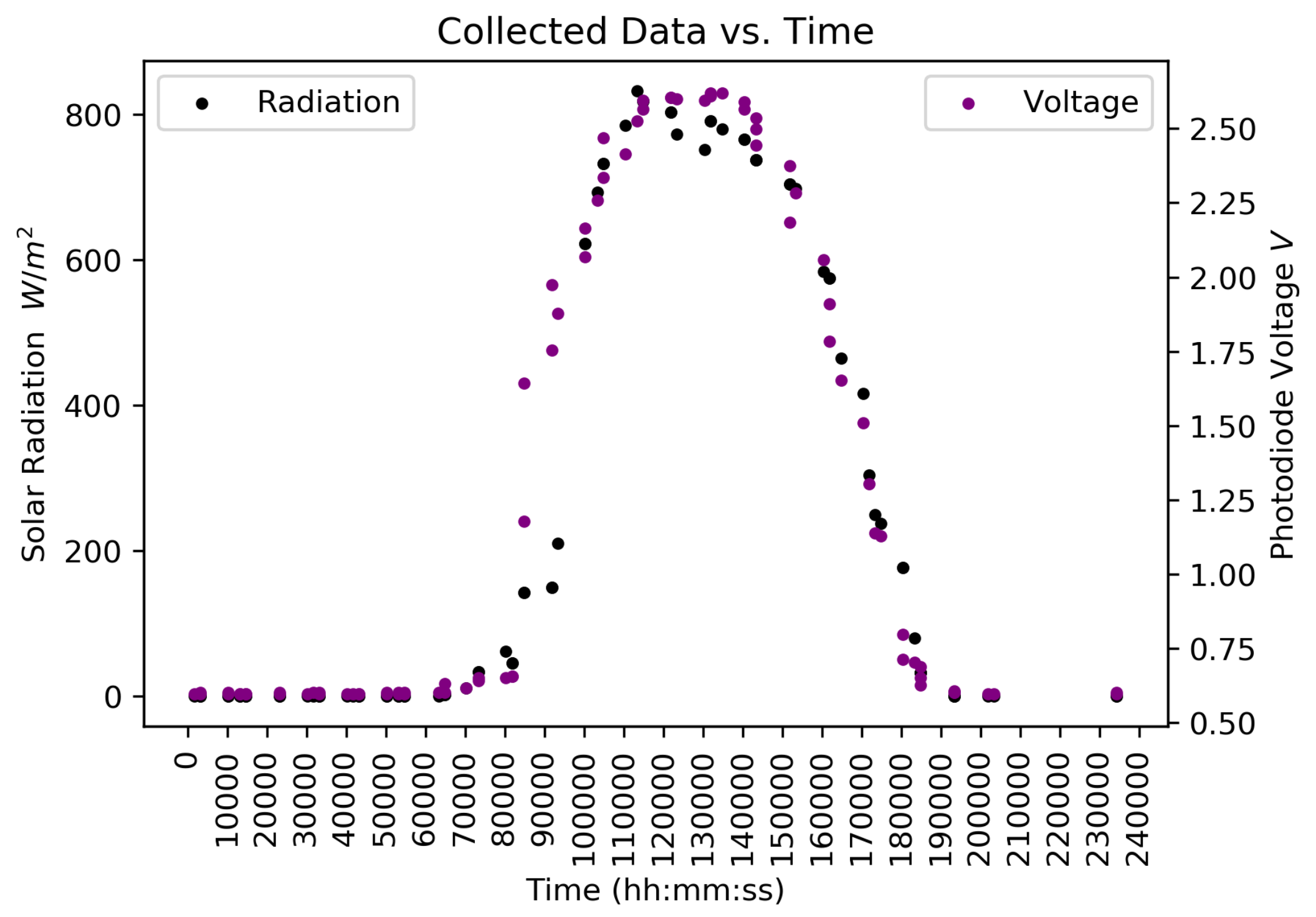
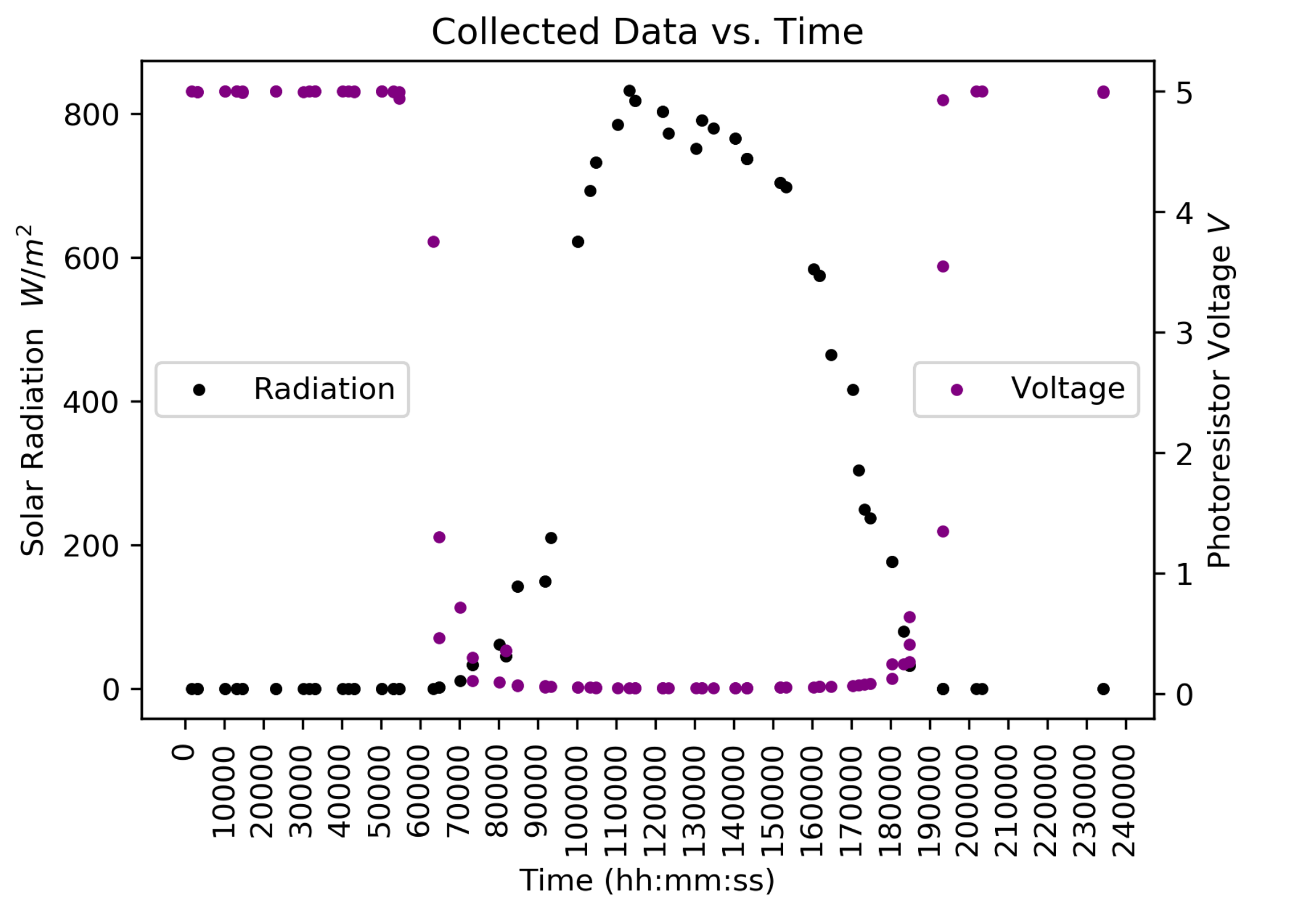


Figure 6


Collected Data vs. Time
Radiation
Voltage
Solar Radiation W/m²
Photoresistor Voltage V
Time (hh:mm:ss)